AIRES: Automatic Integration of Reusable Embedded Systems

Kang G. Shin

Real-Time Computing Laboratory

EECS Department

The University of Michigan

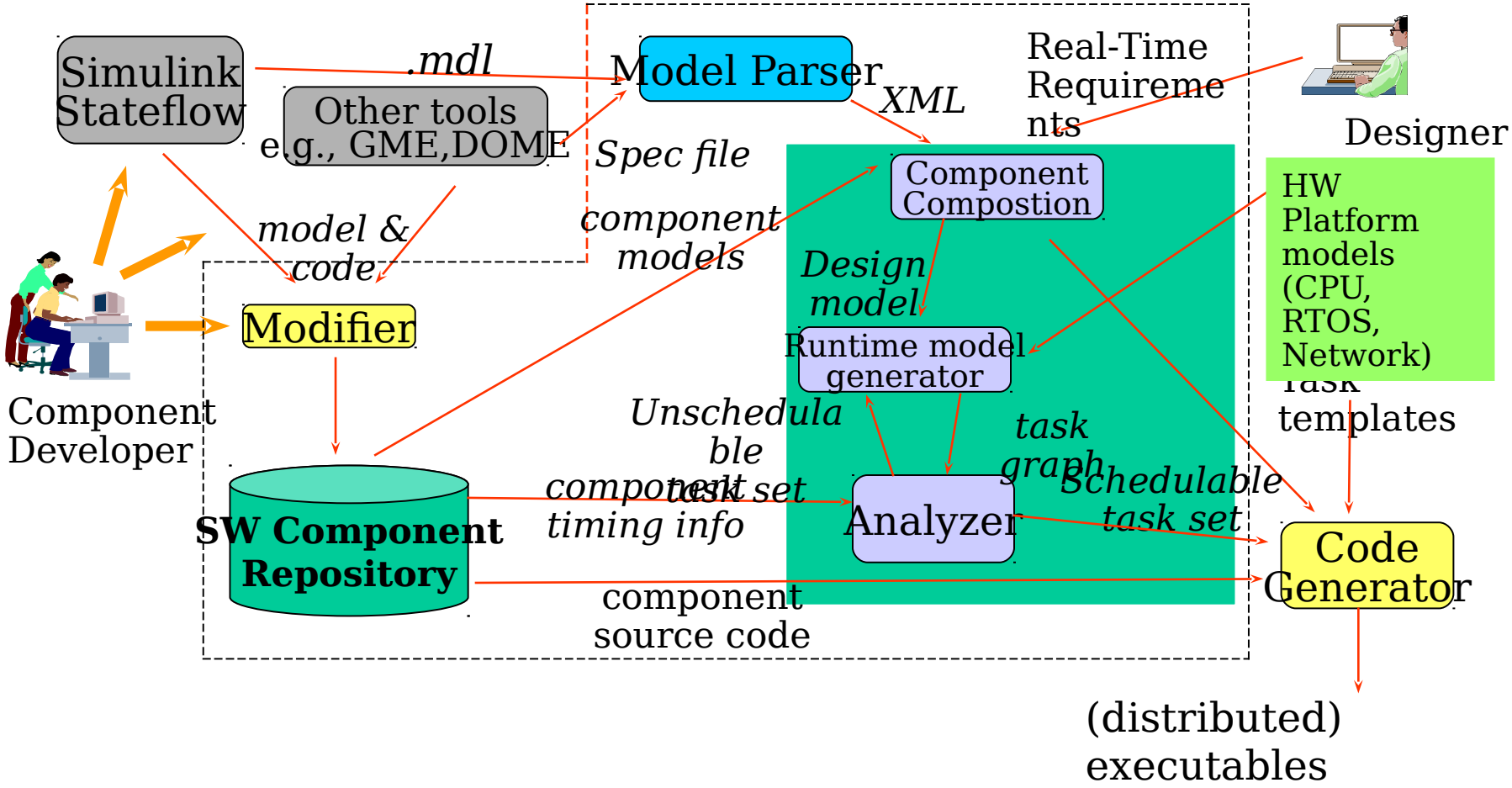
Problem Description

- ***Automatic mapping*** **between views with consideration of real-time performance**
  - **Component composition**
  - **Construction of runtime model from design model**
  - **Assignment of timing and scheduling attributes**
- ***Integration*** **of timing analysis with functional design model and platform information**
  - ***Early-phase*** **performance analysis**
  - ***Performance-aware*** **system design &**

Program Objective

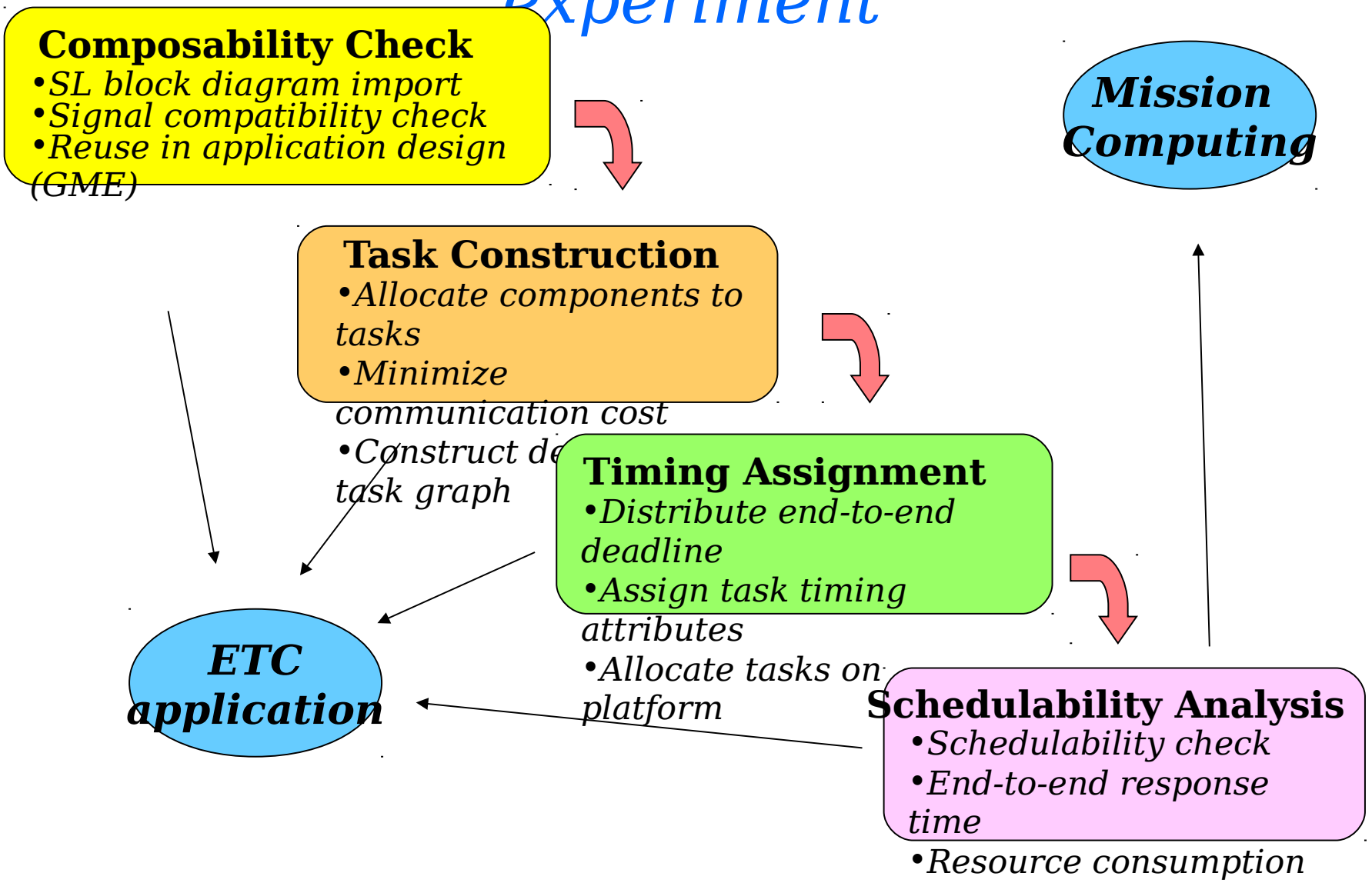
- **Develop algs & toolkit to support:**
  - **Functional composition of reusable components**
  - **Automatic generation of runtime model**
  - **Timing specification and assignments**
  - **Schedulability and timing analysis**
- **Success criteria:**
  - **Toolkit works with various design models (both OEPs)**
  - **Timing constraints can be specified in design models**
  - **Runtime model can be generated from design model with minimum human intervention**
  - **Decisions generated by toolkit result in**

Toolkit Description
Overview



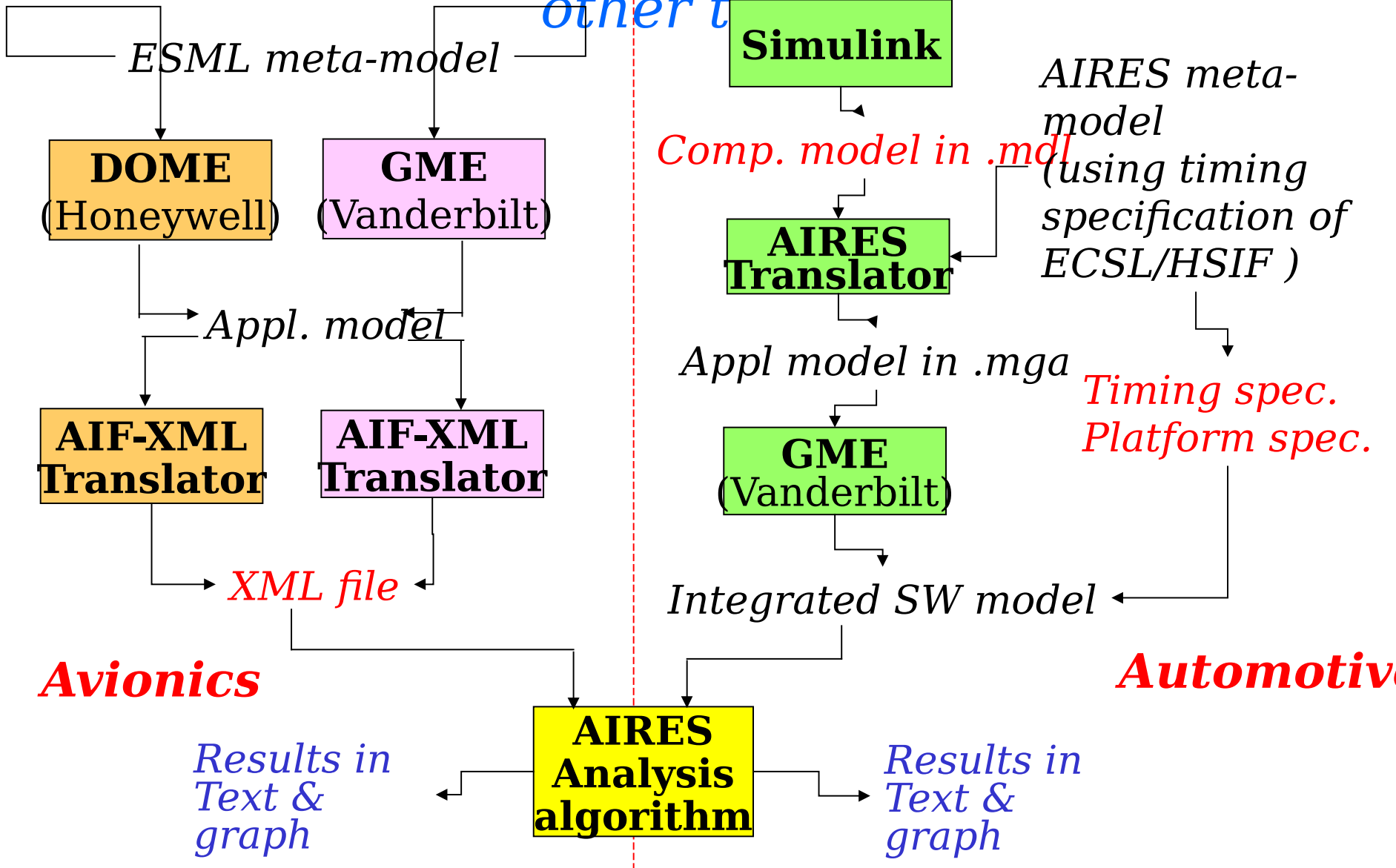
Tool Description

before midterm experiment



Tool Description

interfaces with other tools

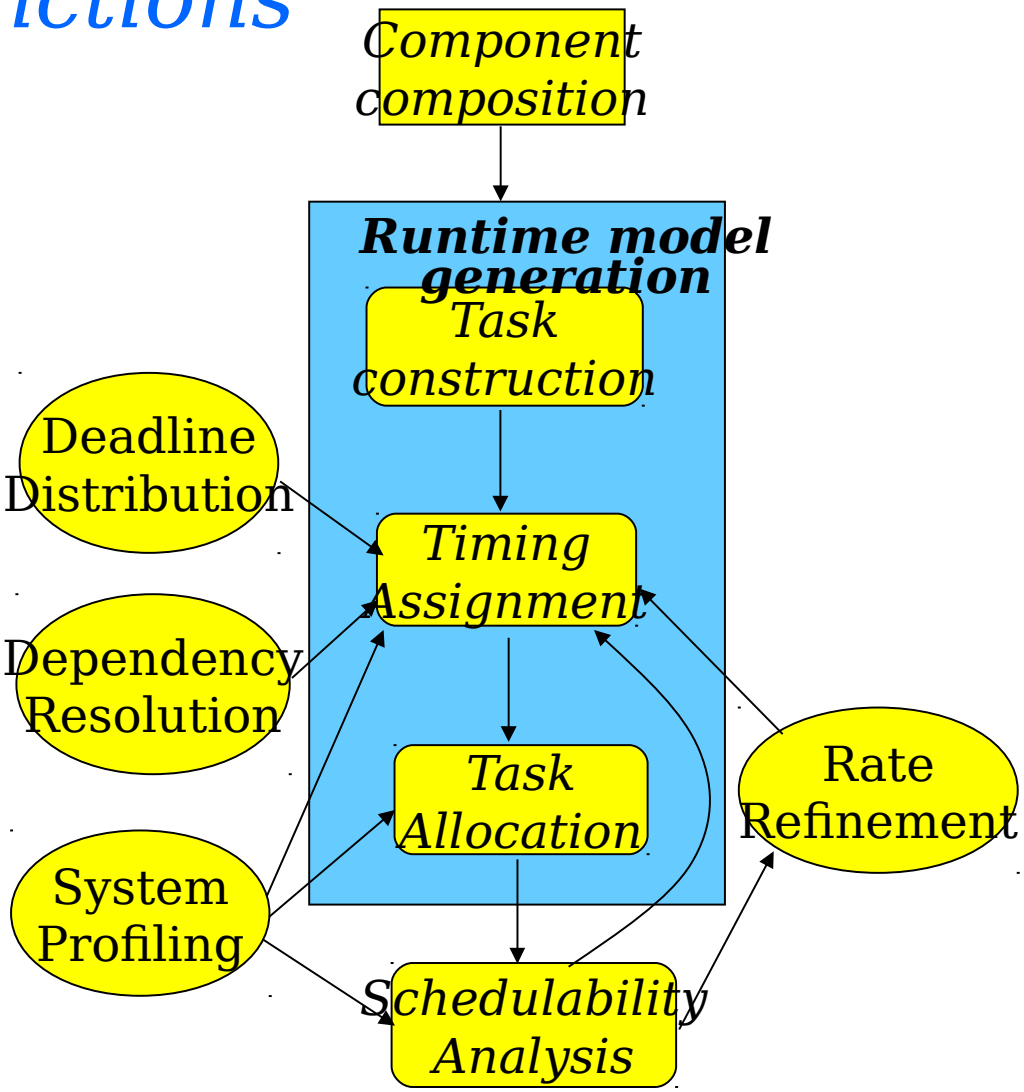


Tool Description *composability check*

- **Existing function**
  - **AIRES meta-model includes task graph, platform and timing specifications**
  - **Component repository**
  - **Signal composability check**
- **Improvement after Midterm Experiment release**
  - **Extended signal composability check: value boundary (e.g., contain, overlap, etc)**

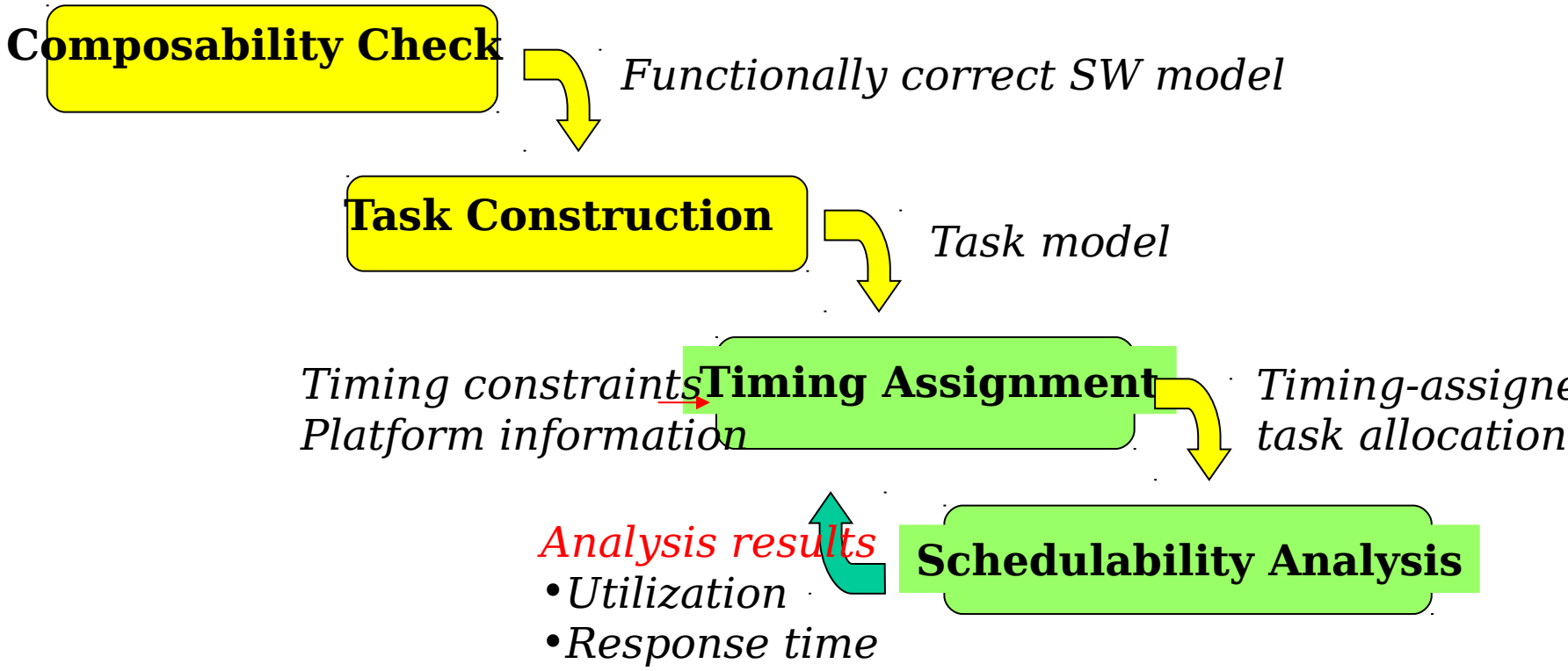
Tool Description *improved functions*

- **Focused on integration of design model, runtime model, and platform configuration**
- **Extended signal consistency check**
- **Added platform info into analysis**
- **Added rate refinement and dependency resolution**



Tool Description
automatic design refinement

- **Feed schedulability/timing analysis results back to design:** first iteration re-assigns timing attributes



OEP Participation

- **Automotive OEP**
  - **Component composability**
  - **Multi-view mapping (level 2 view → level 3 view)**
    - **level 2: discrete-time controllers and some scheduling info**
    - **level 3: platform-specific info**
  - **Task allocation to distributed platform**
  - **Timing assignment and schedulability analyses**
- **Avionics OEP**
  - **Event dependency analysis**
  - **Timing analysis**

OEP Participation

Midterm Experiment Feedback

- **Avionics OEP**
  - **POC:**
    - **Mark Shult: Vanderbilt**
    - **Wendy Roll: Honeywell**
  - **AIRES tool detected errors successfully!**
    - **Event-dependency cycles**
    - **Frame overrun**
    - **Unschedulable task set**
  - **Provides info on component's CPU consumption**
  - **Midterm only test *part* of AIRES tool functionality (due to the *single* processor scenario used)**
  - **May need to detect trapped and unused resources**

OEP Participation

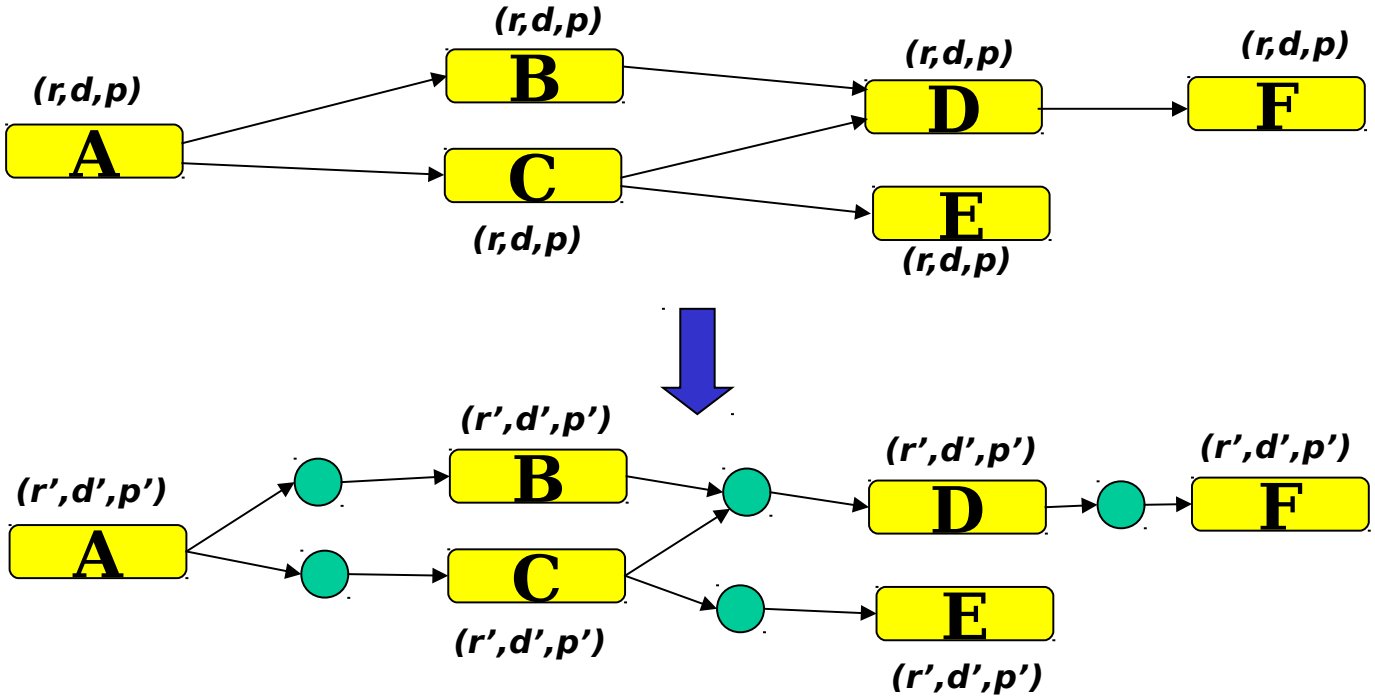
Midterm Experiment Feedback

- **Automotive OEP (Berkeley and New Eagle)**
  - **POCs: Mark Wilcutts at UCB, Scott Ranville at NE-GM**
  - **Our contributions**
    - **Addressed challenge problems:**
      - *Multi-view modeling, model composition (only composability check)*
      - *Schedulability analysis*
      - *Allocation of functions to distributed platform*
    - **Analysis results helped *refine* design**
    - **Provided techniques not supported by commercial tools:**
      - **Automatic task generation and timing assignment**
      - ***Platform-specific* information is integrated**
  - **Communication issues (UCB)**
    - **"Task" has different meanings to different folks.**
    - **Need to improve readability of manuals and documentation**
  - **More modeling improvement with platform info**

Project Status

dependency resolution

- **Goal: support *scalable* task assignment and scheduling algorithms**
- **Approach: *shared buffers* to break dependencies**
- **Issues:**
  - **Derivation of polling rates**
  - **Reduction of polling overhead**



Project Status

dependency resolution algorithm

- **Polling rate:**
  - For each task, the following equation has to be satisfied

$$p_T + r_T \leq d_T - s_T$$

where p_T = the inter- polling interval for T, r_T = T's response time, d_T = T's deadline relative to beginning of task chain, and s_T = T's ready time

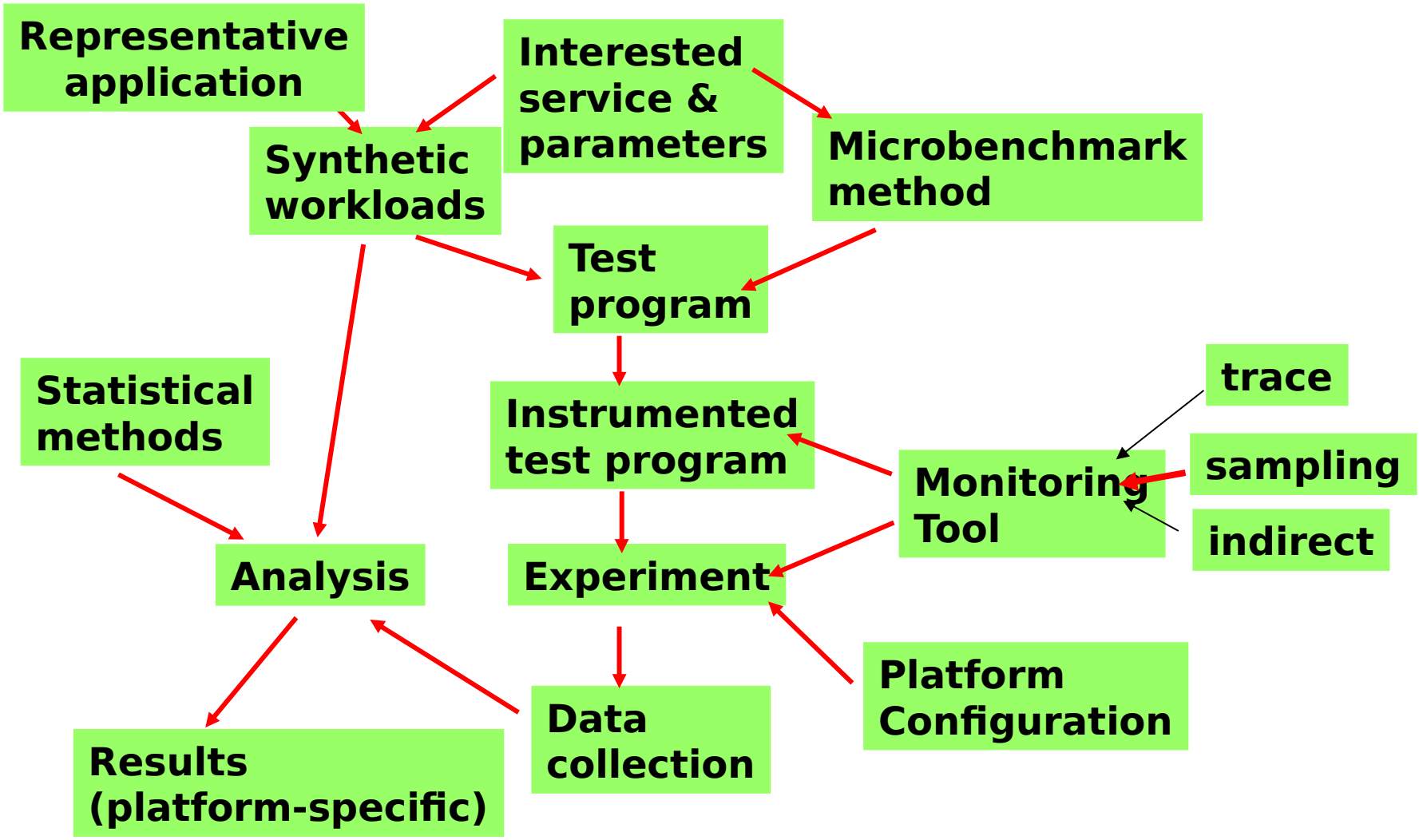
$$s_T = \max\{(p_X + r_X) : X \text{ is immediate predecessor of } T\}$$

- **Polling overhead reduction**
  - Clustering tasks on same CPU and/or using direct synchronization
- **Method is more suitable for event-based scheduler (OSEKWorks, RTLinux)**
- **Better scalability**

Project Status
System profiling

- **Goal**:
  - ***Integrate the effects of target platform in design-time analysis***
  - **Determine platform performance systematically**
- **Approach**: **end-to-end system measurement**
  - **Synthetic workloads**
  - **Microbenchmark**
  - **Sampling tool**
- **Issues**:
  - **Reuse measurement results**

Project Status
profiling method



Project Status

Refinement of timing assignment

Assumptions:

- Only to meet *e2e deadlines*
- *Independent* task set by using shared buffers

Basic idea:

- Consider 3 cases:
  1. *schedulable+constraint met:* done
  2. *schedulable+constraint not met:* enough resource, adjust rate assignments
  3. *Unschedulable:* not enough resource, alter task rates
- When altering rates, evaluate difference between new and original rates for all tasks with higher

Algorithm *Refine*

input: analysis results (utilization, response time), task set, timing constraints

output: task set with new assigned rates

begin

case 1: DONE;

case 2: return to designer to alter constraint or implementation);

case 3:

select the first unschedulable task x;

compute diff(x)=resp(x)-d(x);

for (each task t with higher priority)

compute new r'(t) given resp(x)=d(x);

compute diff(t)=r(t)-r'(t);

choose task s to change with diff(s)=min(diff(t,s,x))

Project Status

Progress and deliverables

- **Progress (since last PI meeting)**
  - **Improved composability with value boundary checks**
  - **Developed algorithm to *resolve task dependencies***
  - **Integrated platform-profiled information in timing assignment and analysis *and* pure priority-based algorithm in analysis**
  - **Developing an algorithm to automatically *refine* timing assignment by feeding analysis results back**
- **Deliverables**
  - **Tools that implement all algorithms in**

Project Status

accomplished milestones

- **Verify semantic correctness of functional composition**
  - **Signal check**
- **Integrate timing specification with functional model**
- **Support runtime model construction**
  - **Deadline distribution for timing assignment**
- **Integrate platform information in analysis**
  - **System profiling**
  - **Schedulability and timing analysis**
- **Feed analysis results back for design refinement**
- **Evaluated with simple OEP scenarios**

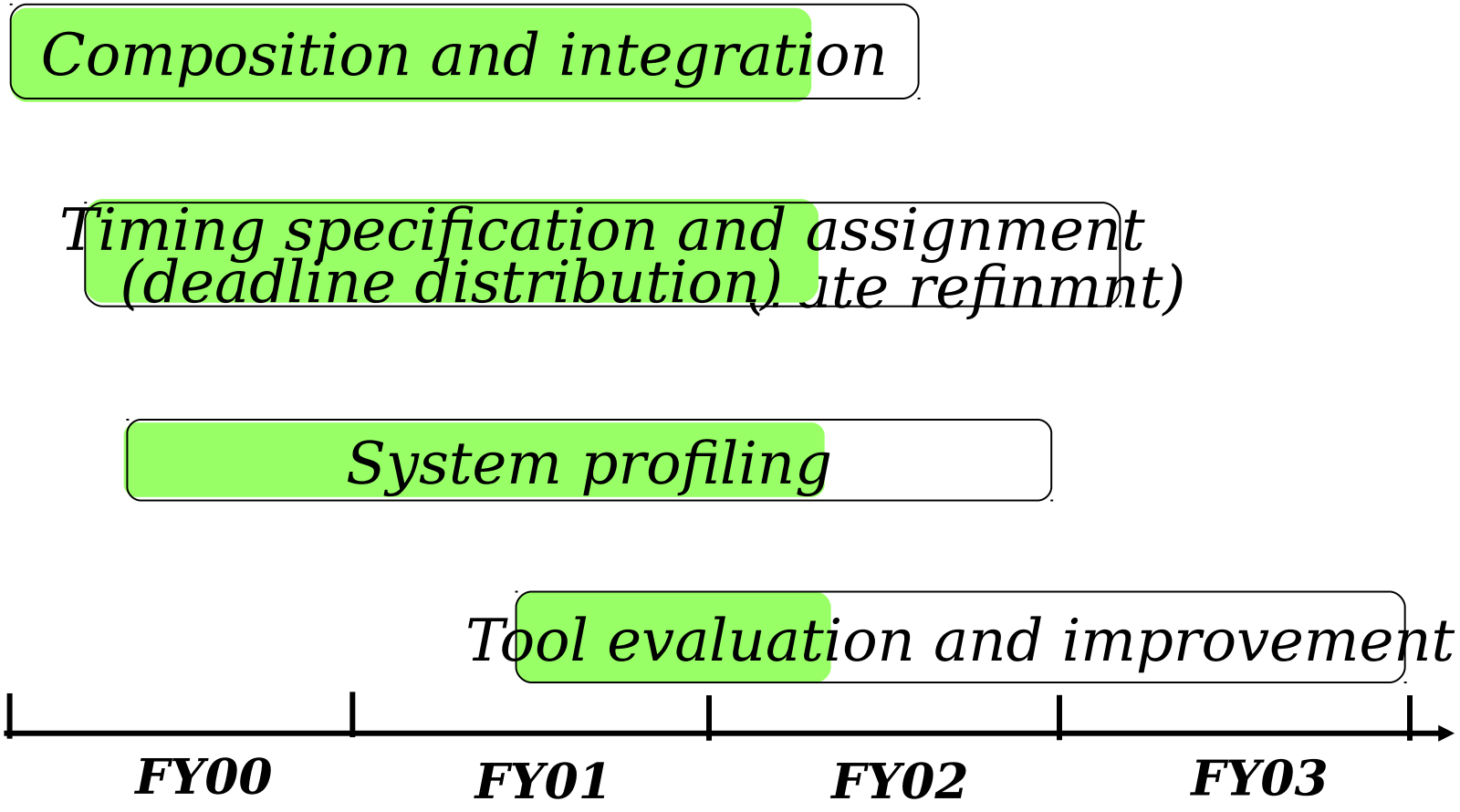
Project Status
Publications

- S. Wang and K. G. Shin, “Constructing reconfigurable software for machine tool control system”, *IEEE Trans. On Robotics and Automation* (in press).
- S. Wang, S. Kodase, K. G. Shin, D. L. Kiskis, “Measurement of OS services and its applciation to performance modeling and analysis of integrated embedded software”, *IEEE RTAS 2002*
- Zonghua Gu and K. G. Shin, ``Analysis of event-driven real-time systems with time petri nets: A translation-based approach,’’ DIPES 2002 Conference (IFIP World Congress), August 2002 (in press).
- S. Kodase, S. Wang, and K. G. Shin, “Resolving

Project Plan

- **Implement and evaluate the dependency resolution for dependent task sets (6 mo)**
- **Develop algorithms to analyze system with mixed scheduling policies and communication delays (in platform model) (6 mo)**
- **Develop method to refine task generation with analysis results when refinement of timing assignment can't meet the constraints (6 mo)**
- **Collaborate with OEP/local industry to**

Project Schedule and Milestones



Technology Transfer and Industry Interactions

- **Interaction with other PIs:**
  - **Provide toolkits/algorithms and documentations**
  - **Interact with OEPs and other industry PIs (e.g., Honeywell) on using AIRES tool**
- **Interaction with local automotive industry:**
  - **Collaborate with GM on engine control software components: ring architecture components**
  - **Use AIRES tool to generate and analyze a scheduler in current GM engine software design**
- **Other interactions with:**
  - **New Eagle on tool improvement and**

Program Issue

- None

END